transform the first instance INSTANCE1 into the first object data ODATA1 in operation S213. The controller 320 may program the first object data ODATA1 to at least one non-volatile memory device NVM included in the first cluster 341 in operation S215.

[0077] During a read operation, the controller 320 may receive the read command RCMD output from the host 200 in operation S231 and may read the first object data ODATA1 from at least one non-volatile memory device NVM included in the first cluster 341 in response to the read command RCMD in operation S233. The controller 320 may transform the first object data ODATA1 into the first instance INSTANCE1 in operation S235. The controller 320 may transmit the first instance INSTANCE1 to the host 200 in operation S237.

[0078] When the controller 320 includes a first CPU related to the communication with the host 200 and a second CPU related to the control of at least one non-volatile memory device NVM included in the first cluster 341, operation S213 of transforming the first instance INSTANCE1 into the first object data ODATA1 and operation S235 of transforming the first object data ODATA1 into the first instance INSTANCE1 may be performed by an API run in the second CPU.

[0079] FIG. 8 is a block diagram of a data processing system 400 according to an exemplary embodiment. Referring to FIGS. 1 through 8, the data processing system 400 may include a plurality of client computers 401-1 through 401-*k* (where "k" is a natural number of at least 3), a network 410, a server 420, and a storage 430. Each of the client computers 401-1 through 401-*k* may be implemented as a PC or a mobile device. The storage 430 may include at least one data storage device 300.

[0080] The data processing system 400 may be a DAS system and the at least one data storage device 300 may be a DAS. The client computers 401-1 through 401-*k* may be connected to the server 420 via the network 410. The server 420 may control a data write operation and a data read operation of the storage 430. The server 420 may be directly connected with the storage 430.

[0081] FIG. 9 is a block diagram of a data processing system 500 according to an exemplary embodiment. Referring to FIGS. 1 through 7 and FIG. 9, the data processing system 500 may include a plurality of client computers 501-1 through 501-*k*, a network 510, a file server 520, and a storage 530. Each of the client computers 501-1 through 501-*k* may be implemented as a PC or a mobile device. The storage 530 may include at least one data storage device 300.

[0082] The data processing system 500 may be a NAS system and the at least one data storage device 300 may be a NAS. The client computers 501-1 through 501-*k* may access the storage 530 via the file server 520 connected to the network 510.

[0083] FIG. 10 is a block diagram of a data processing system 600 according to further embodiments of the inventive concept. Referring to FIGS. 1 through 7 and FIG. 10, the data processing system 600 may include a plurality of client computers 601-1 through 601-*k*, a local area network (LAN) 610, a plurality of servers 620-1 through 620-*s* (where "s" is a natural number of at least 3), a switch 630, and storages 630, 650, and 660. Each of the client computers 601-1 through 601-*k* may be implemented as a PC or a mobile device. The storages 630, 650, and 660 each may include at least one data storage device 300.

[0084] The data processing system 600 may be a SAN system and the at least one data storage device 300 may be a storage that can be used in a NAS. Each of the client computers 601-1 through 601-*k* may be connected to at least one of the servers 620-1 through 620-*s* via the LAN 610. Each of the servers 620-1 through 620-*s* may be connected to at least one of the storages 630, 650, and 660 through the switch 630.

[0085] As described above, according to an exemplary embodiment, a data storage device stores an instance in object-oriented programming language as it is in a memory device embedded therein, thereby reducing or dispersing a calculation load occurring during a process of writing data to the memory device and reading data from the memory device. In addition, the data storage device performs transformation from an instance in object-oriented programming language into object data and from object data into an instance therewithin. The data storage device directly stores an instance in object-oriented programming language in its embedded memory device.

[0086] While the inventive concept has been particularly shown and described with reference to exemplary embodiments thereof, it will be understood by those of ordinary skill in the art that various changes in forms and details may be made therein without departing from the spirit and scope of the inventive concept as defined by the following claims.

What is claimed is:

1. A method of operating a data storage device which is connected to a host and comprises a memory device and a controller, the method comprising:
- receiving, by the controller, a first instance in object-oriented programming language in response to a write command from the host;
- transforming, by the controller, the first instance into first object data; and
- programming, by the controller, the first object data into the memory device.

2. The method of claim 1, further comprising:
- reading, by the controller, the first object data from the memory device in response to a read command from the host;
- transforming, by the controller, the first object data into the first instance; and
- transmitting, by the controller, the first instance to the host.

3. The method of claim 2, wherein the data storage device is a solid state drive and the memory device is identified by one of channels and one of ways.

4. The method of claim 2, wherein the controller comprises a first central processing unit (CPU) in communication with the host and a second CPU in control of the memory device, and
- wherein the second CPU comprises an application programming interface (API) to perform the transforming the first instance into the first object data and the transforming the first object data into the first instance.

5. The method of claim 4, wherein the first CPU and the second CPU share a semiconductor substrate with each other.

6. The method of claim 4, wherein the first CPU and the second CPU are formed in different chips respectively.

7. The method of claim 1, wherein the data storage device further comprises dynamic random access memory (DRAM) connected to the controller, and